REMARKS

This paper is responsive to the Office Action mailed May 22, 2007. Claims 1-51 are currently pending. Claims 1-5, 7, 17-37, 39, 49 and 50 have been amended. No new matter has been added.

At paragraphs 3 and 4 of the Office Action, the Examiner objects to the drawings as failing to comply with 37 CFR 1.84(p)(5). The Applicant has amended the specification to overcome the Examiner's objections. Accordingly, those objections should be withdrawn.

At paragraph 7, the Examiner rejects claim 49 under 35 U.S.C. §101 as being directed to non-statutory subject matter. The Applicant amends claim 49 to overcome the Examiner's rejection. Claim 49 as amended is directed to a program storage medium, and as such is directed to statutory subject matter. That rejection should be withdrawn.

At paragraph 9, the Examiner rejects claims 1-51 under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim subject matter which the Applicant regards as his invention. Accordingly, the Applicant amends claims 1-3, 5, 7, 17-19, 21, 23, 33-35, 37, 39, 49 and 50 for clarity, to be consistent with the Examiner's suggestions. Accordingly, those rejections should be withdrawn.

At paragraph 11, the Examiner rejects claims 1-4, 6-9, 11, 13-15, 17-20, 22-25, 27, 29-31, 33-36, 38-41, 43, 45-47 and 49 under 35 U.S.C. §102 as being anticipated by U.S. Patent No. 6,820,255 (Babaian). Independent claims 1, 17 & 33 have been amended to more clearly distinguish the present invention over the cited prior art of Babaian.

Babaian discloses a binary translation process which builds a database of translated binary code and then checks for correspondence between the foreign code and the translated binary code stored in the database. If the database contains corresponding code, then that already translated code is transferred to system memory for execution, thus avoiding the need to retranslate the foreign code. This understanding of Babaian is supported by the referenced portions of columns 4 to 9 mentioned in paragraph 11 of the Office Action.

Babaian is a good example of the prior art which provides a "local cache" to store binary code translations produced by the translator when translating from foreign binary code to equivalent host code. However, Babaian does not disclose providing a "first translator instance" and a "second translator instance" as recited in claims 1, 17, 33 & 49. Further, Babaian does not disclose that the first translator instance translates and caches a portion of the target code and that the second translator instance "retrieves the cached portion of the target code" when translating the subject code into the target code as recited in claims 1, 17, 33 & 49. Further still, Babaian does not disclose that the second translator instance retrieves the cached portion of the target code "upon a compatibility detection between said portion of the target code and a second portion of the subject code" as recited in claims 1, 17, 33 & 49.

At paragraph 12, the Examiner rejects certain dependent claims under 35 U.S.C. 103(a). In light of the remarks above regarding Babaian and the independent base claims from which those dependent claims depend, the dependent claims rejected under 35 U.S.C. 103(a) should be allowable. The additional referenced cited in the 35 U.S.C. 103(a) rejections do not supply that which is missing from Babaian.

The described embodiments are a significant improvement over Babaian, whether this document is read alone or taken in combination with any of the other citations such as Curtis, Ronstrom, Miller or Nelson mentioned in the Office Action. Importantly, the described embodiments provide first and second translator instances which both translate the subject code into the target code. As explained for example at paragraph [0194] of the detailed description, the described embodiments allows the second translator instance to benefit from the efforts of the first translator instance in creating the cached portions of target code, particularly when the second translator instance is translating subject code from the same subject program as the first translator instance or from a different program that has common portions of subject code (e.g. common system libraries). The described embodiments allow the portions of target code and related translation structures that were created by the first translator instance to be reused, when compatibility is determined, by the second translator instance which encounters compatible subject code portions. However, there are severe obstacles to using target code produced by a first translator instance in a second translator instance. It is not a straightforward or routine matter to share the target code produced by the first translator instance with the second translator instance. These further obstacles are overcome by the described embodiments.

For example, the target code is often created and optimized by the first translator instance according to the set of conditions encountered when the target code was generated from the first section of subject code. That is, the target code produced by the first translator instance is not necessarily compatible with the needs of the second translator instance with respect to the second section of subject code, because the first translator instance may produce target code which is optimized for a specific set of conditions. Further, the target code is often changed and revised by the first translator instance - such as in response to self-modifying code events or after additional optimization of the target code. Hence, in the described embodiments, the method further includes the "compatibility detection" step recited in the independent claims 1, 17, 33 and 49.

As discussed in the detailed description from paragraphs [0198] through [0223] concerning "cache evolution", "parallel translation," "shared memory," and "aggressive optimization," the exemplary embodiment address other specific problems which arise when attempting to share target code portions between first and second translator instances. These features of the exemplary embodiments are also reflected in the new dependent claims 52-67.

In summary, the independent claims are allowable not least because Babaian and the other cited prior art documents do not disclose "providing first and second translator instances," cacheing a target code portion from the first instance for reuse by the second instance, or a compatibility detection by the second instance between the cached portion of target code and a second subject code portion, as recited in the independent claims 1, 17, 33 & 49. The dependent claims are allowable because they recite further novel and inventive features of the invention, and also because the dependent claims each depend from allowable main claims.

Filed herewith is a Request for a Two-Month Extension of Time, which extends the statutory period for response to expire on October 22, 2007. Accordingly, Applicant respectfully submits that this response is being timely filed.

In view of the above amendment, applicant believes the pending application is in condition for allowance. Applicant believes no fee is due with this response. However, if a fee is due, please charge our Deposit Account No. 08-0219, under Order No. 1801270.00140US1 from which the undersigned is authorized to draw.

Respectfully submitted,

Dated: October 19, 2007

[illegible]

Ronald R. Demsher
Registration No.: 42,478
Attorney for Applicant(s)

Wilmer Cutler Pickering Hale and Dorr LLP
60 State Street
Boston, Massachusetts 02109
(617) 526-6000 (telephone)
(617) 526-5000 (facsimile)